Extended Impedance Control Using Real and Virtual Sensors for Redundant Manipulators

Mark K. Long and Paul G. Backes
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, 91109

Abstract

Control of a redundant manipulator based on an impedance control framework with multiple simultaneous control sources is described. Each control source provides a different behavior type. An application is decomposed into multiple simultaneous behaviors whose resultant behavior will provide the motion necessary to execute the task. The simultaneous control inputs are merged using impedance control to compute a resultant command to the manipulator. The task space of each behavior can have the dimensionality of the mechanism being controlled. Control of a seven degree of freedom manipulator is described here with an available task space for each behavior of dimensionality seven.

1. Introduction

This paper describes a method for control of kinematically redundant manipulators for telerobotics applications. This is one of the prototype technologies being developed in the JPL Supervisory Telerobotics (STELER) Laboratory for local/remote telerobotic operations. Telerobotics must provide reliable operations in partially structured and unstructured environments as well as provide a wide range of capabilities with modest computing power. Supervisory Telerobotic systems have a wide variety of applications including maintenance, inspection, experiment tending, and servicing. To achieve these goals the technologies of dextrous manipulation using kinematically redundant arms, active compliant control, and multi-sensor shared control have been brought together in a unified control framework. This single control structure will enable successful completion of tasks while being able to respond to unplanned scenarios, IIIIsIt'11('till'('d (']11'ilc)]IItl('ills and anomalies.

A large number of motion sources may be necessary in a robot control system which is expected to perform a wide variety of tasks. For example, a trajectory generator may be needed to provide position setpoints, force sensor inputs may be necessary for contact applications, hand controller inputs may be needed for operator control, gripper motion for grasps, and visual feedback for automatic alignment. Many tasks require a simultaneous combination of motion sources. A compliant grasp task requires simultaneous force control and gripper control. A shared control polishing task could use hand controller inputs to specify motion tangential to a surface while force control controls the force of contact with the surface and the manipulator reconfigures itself in real time to stay away from joint limits, singularities and collisions.

The necessary number of degrees of freedom (DOFs) of the mechanism can also vary depending on the task. A four DOF scara manipulator is sufficient for many pick and place operations. A six DOF manipulator is sufficient for placing objects in an arbitrary orientation. A seven DOF manipulator provides an ability to continuously change its internal link configuration for a constant tool position and orientation, and can extend the dextrous workspace. For the seven DOF manipulator, the possible dimension of the output motion (seven) is greater than the dimension of possible motion of the gripper (six). All of the mechanism DOFs should be available for task execution. The control scheme must therefore allow both a variable number of simultaneous input sources and a variable dimension task space.

Previous approaches to redundancy resolution include pseudoinverse and augmented (or composite) Jacobian methods. A wide variety of pseudoinverse approaches have utilized projection operators on the null space of the end-effector Jacobian to resolve the redundancy. These methods cannot in general guarantee cyclic or conservative motion and more importantly often cannot guarantee direct control of the entire mechanism. This often results in some unspecified internal motion of the mechanism. These methods also rely heavily on optimization functions to resolve the redundancy with only local results and often with objective functions of questionable utility to the task at hand. Augmented Jacobian methods focus on defining kinematic functions that provide forward and differential kinematic relationships to attempt to fully specify the motion of all degrees of freedom of the mechanism. These approaches have been fairly successful[1, 2, 3] but still have some difficulties. Often the forward kinematic relationships are difficult to compute and may not be defined over the entire workspace. Additionally the majority of these functions do not have closed form Jacobian relationships and require numerical techniques for computation. Many of the problems with artificial singularities of these new kinematic functions and discontinuities in switching from one function or subtask have been addressed using damped least squares for position controlled applications [4, 1, 2]. However, damped least squares with joint velocity weighting introduces tracking error throughout the workspace and requires the selection of weighting matrices which is often non-intuitive. Many of these issues have not been addressed for force or impedance control applications.

The approach to control of kinematically redundant manipulators described here uses a fixed task space parameterization of the same dimension as the robot. This is desirable from the standpoint of directly controlling all degrees of freedom and eliminating uncontrolled internal motions which can pose safety, collision, and cyclicity problems. Additionally it is desirable to make use of all available degrees of freedom to complete kinematically challenging tasks in unstructured environments that may not be part of the planned suite of capabilities.

The task space parameterization does not change over time. Instead an impedance model is applied to the full dimension of the task space and shared control techniques are used to combine real and virtual sensory inputs to the impedance model. In this way, degree(s) of freedom normally referred to as redundant are unified with the task space and reference trajectories as well as sensor data cause motion of the entire mechanism. Additionally since the task space parameterization is fixed, issues of switching discontinuity do not arise and singular regions associated with the task space parameterization are fixed. The entire task space of the manipulator can be used to accomplish tasks as motion is mapped to the appropriate degrees of freedom. Thus online numerical techniques are not required to compute differential (Jacobian) relationships. When operating in these regions potential fields may be applied to the impedance model to inhibit movement in the singular or increasingly singular directions. Alternatively, if position based impedance control is being used, the damped least squares methods can be used with the tracking error producing velocity weighting only active in the singular regions. The parameterization of redundancy discussed here generally have some physical significance, however,

there also exist a wide variety of kinematic arrangements that may not have a "natural" redundancy parameter that has physical significance or spans a significant portion of the workspace. In such cases one can rightly inquire into the motivation behind such a design from a task execution perspective and determine the minimum set of parameters that will completely describe the task space for the task at hand.

There are different ways to implement a system to provide multiple input sources. One solution is to provide a flexible robot programming environment which provides a layered set of subroutines for robot applications programming. A custom program could then be developed to utilize the needed sensors for a specific task. A robot language could also be used to develop a program to merge control based upon multiple sensors. The approach used to implement the control architecture of this paper is to provide a fixed software system with data driven execution. The control system provides a large suite of capabilities based upon input data. This approach is used to satisfy requirements for space telerobotics where the flight component of the telerobotic system must be flight qualified. The fixed flight software can provide the multiple control sources with the behavior of control from each source dependent on the parameterization data which can be sent from a distant ground station. The actual task execution motion is the resultant behavior of all the input sources.

This paper describes a control architecture which allows execution of a task to be considered as the resultant behavior of execution of multiple concurrent behaviors. The dimensionality of the execution space of each behavior and the resultant behavior can be extended to the dimensionality of the controlled mechanism. Task description consists of decomposing the desired execution into the multiple simultaneous behaviors. Each behavior generates motion commands which are merged in a common motion space to compute a resultant command to the manipulator. The task space of each behavior can have the dimensionality of the mechanism being controlled. The architecture is applied to control of a seven DOF manipulator. The results are also applicable to other redundant and non-redundant manipulators with various numbers of DOFs. Previous work has described techniques for compliant motion control [5, 6, 7], shared control [8], and redundancy resolution [9, 10, 11].

2. Control architecture for multiple simultaneous behaviors

The control architecture for simultaneous execution of multiple control behaviors is shown in Figure 1

The *Application Space* includes all potential application tasks which the robot control system must be able to accomplish. These application tasks could be sequenced together to accomplish a larger task. Execution of a given application task can be decomposed into concurrently executing behaviors. For example, a door opening task could utilize a trajectory generator to generate the nominal trajectory while force control adds small perturbations to adjust for errors between the planned trajectory and the physical system motion. The *Command To Behavior Map* performs the mapping between the task and the required concurrent behaviors. This could be done automatically or through interaction with an operator.

The *Behavior Space* includes all of the independent control behaviors. Trajectory Tracking is a control behavior which provides a trajectory generator to generate real-time trajectories. The Teleoperation behavior takes real-time operator inputs and generates control inputs. Dither generates small periodic dither control inputs. Force Tracking provides control of contact forces between the manipulator and the environment. Manipulability computes an optimum arm configuration and generates control inputs to move toward it. Singularity Avoidance generates control inputs to keep the arm away from singularities.

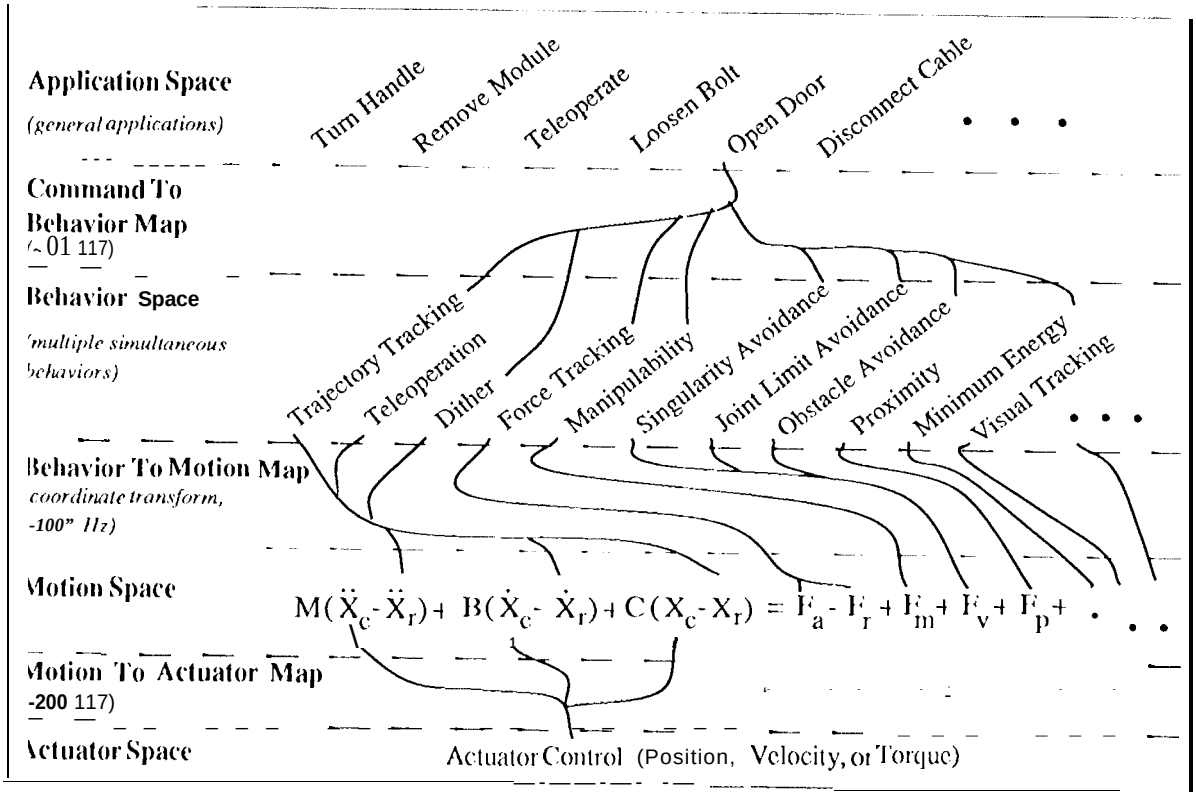

Figure 1: Task decomposition and control for concurrent behaviors

Joint Limit Avoidance generates control inputs to keep the arm away from joint limits. Obstacle Avoidance generates control inputs to prevent collisions. Proximity generates control inputs to control proximity to a real or virtual object. Visual Tracking generates control inputs to provide visual servoing. Other behaviors could also be provided. Each behavior has command parameters that specify its operation and use of real and virtual sensor data. Virtual sensors are those that derive data, possibly from real sensors, e.g., a joint limit or singularity sensor derives data from real joint position sensors.

More complex resultant behaviors can be generated by concurrent execution of individual behaviors. For example a polishing behavior may be composed of teleoperation, force tracking, manipulability, joint limit avoidance, obstacle avoidance, and singularity avoidance behaviors. Teleoperation could allow motion inputs by an operator only tangential to the surface normal. Force tracking could provide a constant force against the surface. Manipulability could control the arm configuration for optimal control of fine forces. Joint limit avoidance, obstacle avoidance and singularity avoidance would keep the arm from collisions and singularities. The operator would then only have to provide the motion over the surface. The autonomous system would provide the rest of the control.

The *Motion Space* is the common control space for all behaviors. Mechanisms with more than six mechanical DOFs have been referred to as kinematically redundant since the classical problem of end-effector position and orientation control for a spatial manipulator can be handled by a six DOF robot. Task requirements often dictate a task space of dimension greater than six. For so called kinematically redundant robots, a motion space is defined that spans all of the mechanical DOFs. The motion space of the seven DOF manipulator used here includes a six DOF coordinate frame (the MOTION frame), and an "arm angle" parameter which describes the internal configuration of the arm. The arm angle, represented by ψ, is defined as the angle between the plane passing through the shoulder, elbow, and wrist points and some reference plane; we chose the vertical plane here.

Each motion DOF can receive inputs from multiple behaviors. Motion Space control is done here using impedance control [5] but with the expanded ability to merge multiple control inputs either as position inputs or force inputs. In addition, the impedance equation can be extended beyond six DOFs to match the dimension of the motion space. The motion space impedance control equation, as shown in Figure 1, is

$$M \cdot (\ddot{X}_c - \ddot{X}_r) + B \cdot (\dot{X}_c - \dot{X}_r) + K \cdot (X_c - X_r) = \sum F_i \tag{1}$$

where M is the inertia matrix, B is the damping matrix, K is the stiffness matrix, X_r is the reference trajectory, X_c is the commanded position, and $\sum F_i$ is the sum of all behavior inputs mapped to forces. Behaviors can also generate position commands to be merged with the reference trajectory. Equation 1 is implemented with

$$\ddot{X}_c^{n+1} = \ddot{X}_r^{n+1} + M^{-1} \cdot [\sum F_i^n - B \cdot (\dot{X}_c^n - \dot{X}_r^{n+1}) - K \cdot (X_c^n - X_r^{n+1})] \tag{2}$$

$$\dot{X}_c^{n+1} = \dot{X}_c^n + \ddot{X}_c^{n+1} \cdot \Delta t \tag{3}$$

where n is the current sample number and Δt is the sample time between samples.

This gives the desired acceleration of the mechanism in the Motion Space which is integrated to produce the desired velocity. The motion commands are then mapped into the actuator space of the mechanism.

The *Actuator Space* is defined as the space of active actuation of the mechanism. Mechanisms which have more than six actuator DOF's fall into two general categories, kinematically redundant and

actuationally redundant. Typically, kinematically redundant mechanisms have additional behaviors associated with position and actuationally redundant manipulators have additional behaviors associated with force [9]. For most applications the motion space should completely span the actuator space of the manipulator to provide the widest array of behaviors for task execution. The mapping is then one-to-one and common Jacobian transpose and Jacobian inverse techniques apply. If there are more DOFs in the actuator space than in the motion space, the mapping is underconstrained and a variety of techniques can be used including pseudo-inverse or minimum kinetic energy [9]. Conversely, if there are fewer DOFs in the actuator space than the motion space, the problem is overconstrained and damped least-squares [12, 13] and other techniques are available. Care must be taken to assure that a one-to-one mapping between motion space and actuator space does not degenerate at or near a singularity.

Although Jacobian inverse routines could be used, a damped-least squares inverse is used here to allow further task prioritization and singularity robustness [12, 13, 10, 14, 11]. The motion space velocity vector of the manipulator has three translational coordinates, three orientation coordinates and the arm angle. A composite Jacobian is formed from the individual Jacobians that relate the rate of change of the joint angles to the rate of change of the motion space parameters. Here the composite Jacobian, J^C, is given by [11]:

$$J^C_{7\times 7} = \begin{pmatrix} J^\omega_{3\times 7} \\ J^v_{3\times 7} \\ J^\psi_{1\times 7} \end{pmatrix} \tag{4}$$

where J^ω is the angular velocity Jacobian, J^v is the linear velocity Jacobian, and J^ψ is the arm angle Jacobian. J^ω and J^v are readily available using:

$$\begin{pmatrix} J^\omega_r \\ J^v_r \end{pmatrix} = \begin{matrix} \mathbf{z}_1 & \mathbf{z}_2 & \cdots & \mathbf{z}_7 \\ \mathbf{z}_1 \times \mathbf{P}_{1,r} & \mathbf{z}_2 \times \mathbf{P}_{2,r} & \cdots & \mathbf{z}_7 \times \mathbf{P}_{7,r} \end{matrix} \tag{5}$$

where r is the velocity reference point, $\mathbf{z}_i$ is the z axis of joint i, and $\mathbf{P}_{i,r}$ is the position vector from the ith link frame to the velocity reference point r. The arm angle Jacobian is available from [11]:

$$J^\psi = \frac{(\hat{\mathbf{w}}\times\mathbf{p})^T}{\|\mathbf{p}\|^2}\mathbf{E} + \frac{\hat{\mathbf{V}}_T^T\mathbf{w}}{\|\mathbf{h}\|^2}(\hat{\mathbf{w}}\times\mathbf{h})^T - \frac{\hat{\mathbf{w}}^T\mathbf{e}}{\|\mathbf{w}\|\|\mathbf{p}\|^2}(\hat{\mathbf{w}}\times\quad)^T\quad\mathbf{W} \tag{6}$$

where $\mathbf{w} = \mathbf{P}_{0,7}$, $\mathbf{e} = \mathbf{P}_{0,4}$, $\mathbf{p} = \mathbf{e} - \hat{\mathbf{w}}(\hat{\mathbf{w}}^T\mathbf{e})$, $\mathbf{V}_T$ is the vector specifying the reference plane, $\mathbf{h} = (\mathbf{w}\times\mathbf{V})\times\mathbf{w}$, $\mathbf{E}$ is the elbow linear velocity Jacobian, and $\mathbf{W}$ is the wrist linear velocity Jacobian. Notice that most of the required data for J^C is available as a by product from a forward kinematic iteration [11].

With the motion space command vector, $\dot{X}_c$, and the motion to actuator space map, J^C (henceforth written as plain J), the joint servo velocity commands can be computed using damped-least squares with

$$\dot{\theta}_d = [J^T\cdot W_T\cdot J + W_V]^{-1}\cdot J^T\cdot W_T\cdot\dot{X}_c \tag{7}$$

where W_T is a diagonal task weighting matrix that relates the relative priorities of the tasks. W_V is a diagonal velocity weighting matrix which weights the norm of joint velocities. It is important to note that while a non-zero W_V matrix will provide robustness to singularities by limiting excess joint velocities, it will also induce tracking error over the entire workspace. By setting W_T to identity and W_V to zero a standard inverse Jacobian result is provided with the same algorithm. The present

implementation utilizes joint position servos so the joint velocity commands are integrated to generate the joint position commands.

3. Individual behaviors

As shown in Figure 1, various individual behaviors can execute concurrently. A behavior can generate either position commands, which are merged with the reference position trajectory on the left side of Equation 1, or force commands which are merged with $\sum F_i$ on the right hand side of the equation.

Teleoperation is shown as a position based input in Figure 1 but is implemented as a force based input, related to input velocities, here. The input velocity motion by the operator with a six DOF hand controller is transformed to equivalent velocities, $\dot{X}_h$, based upon the selected teleoperation mode [15]. These velocities are multiplied by a damping matrix, B_t, to compute the equivalent forces with

$$F_t = B_t \cdot \dot{X}_h \tag{8}$$

The damping matrix, B_t, can be used to select operator input directions. The operator inputs arm angle velocity by pressing a trigger on the hand controller and changes the sign by pressing a button on the hand controller.

Forces are often not controlled directly with impedance control. Rather, a position setpoint is specified inside an object and the actual steady state applied force is a function of both the target stiffness and the position error. This approach is available with this implementation, but an alternative approach has also been implemented. In the alternative approach, a reference (desired) force is specified and the difference between the reference and actual forces is used on the right side of Equation 1. Then exact force control is possible by setting the reference stiffness, K, and the reference trajectory velocity and acceleration to zero in force controlled DOF's. Assuming that the environment can be modeled as a stiffness with spring constant k_{env}, the applied force in a DOF will be

$$f_a = k_{env}(x_r - x_e) \tag{9}$$

where x_r is the position at the initial contact point. If $x_e = x_e - x_r$, then the impedance equation (with no stiffness) for this DOF is

$$m\ddot{x}_e + b\dot{x}_e = -k_{env}x_e - f_r \tag{10}$$

In steady state the desired and actual applied forces will be equal for any target impedance parameters which provide stable contact for the characteristic equation

$$ms^2 + bs + k_{env} = 0 \tag{11}$$

where m is the mass term of M, D is the damping term of B, and k_{env} is the stiffness in the force controlled DOF. Either approach to control of forces is available if the difference between the reference and actual forces is added to the right hand side of Equation 1. This is shown in Figure 1 with the difference $F_a - F_r$.

The trajectory generator behavior computes the reference trajectory acceleration $\ddot{X}_r$, velocity $\dot{X}_r$, and position X_r [16]. The arm angle is also generated as part of the trajectory. An alternative trajectory generation scheme has also been implemented. Here $\ddot{X}_r = \dot{X}_r = 0$ and the reference position is set to be the desired final position. The stiffness term of the impedance equation then causes the arm to move to the destination. A slight drawback with the "spring" trajectory is that the motion accelerates quickly initially and then approaches the destination slowly.

Joint travel limiting provides an artificial potential field at the end of travel limits on each joint. This field is then mapped to the motion space to resist operator commands that exceed joint limits. While the local site path planner can predict and avoid joint limits in its commands, often the operator using teleoperation cannot. The joint travel limiting sensor resists this motion so the operator does not induce a fault condition which would interrupt the current task. Similar to the joint travel limiting behavior is the behavior which limits motion in the manipulator workspace singular regions. Information in joint space or motion space about the singular regions is required. Some singular regions are qualitatively located at joint limits; these are taken care of by the above behavior. Others are located at configurations when the seventh joint frame is within 0.2 meters or beyond 1.1 meters of the first joint frame [17]. Thus if $\|{}^0P_7\| > 1.1$ meters then

$$F_{singularity} = -({}^{motion}_{7}T_f) \cdot K_{singularity} \cdot (\|P_{actual}\| - 1.1) \cdot P_{actual} \quad (12)$$

or if $\|{}^0P_7\| < 0.2$ meters then

$$F_{singularity} = -({}^{motion}_{7}T_f) \; K_{singularity} \; (0.2 - \|P_{actual}\|) \; P_{actual} \quad (13)$$

where $K_{singularity}$ is the gain for the singular region avoidance, ${}^{motion}_{7}T_f$ is the rigid body transformation between the joint 7 frame and the motion space coordinate system, P_{actual} is the actual current position of the joint seven frame, and $F_{singularity}$ is the singular region avoidance behavior command in the motion space. Note that if the manipulator is not near a singular region there is no commanded motion from this behavior.

Other behaviors can also be added either as position inputs or force inputs. The trajectory generator, force control, and teleoperation behaviors have been implemented. Others are planned for implementation.

4. Bounded behavior execution

The control scheme for concurrent behaviors merging has been developed for space flight applications. Therefore it has been implemented with a fixed software system as described below in Section 5. An additional feature which is necessary for execution of tasks in a remote space environment is bounded behavior control execution [18]. The multiple concurrent behaviors are merged together to generate the resultant behavior. This resultant behavior must then be monitored during execution to make sure that it stays within specified bounds for safety. The local site can plan tasks and simulate the execution on a local simulator, but cannot be sure of the motion generated by real-time sensor based motion. To ensure safety the local site can specify and verify safety of tasks which execute within specified bounds. These bounds may include the difference between the reference trajectory and the actual trajectory, force thresholds, and proximity thresholds. As long as the execution progresses within the specified bounds, it should be safe.

5. Laboratory implementation

The data driven merging of concurrent behaviors for a redundant manipulator has been developed for control of Space Station manipulators. The development and implementation has been done in the JPL Supervisory Telerobotics (STELER) laboratory. The STELER lab telerobot system is composed of a local site where task commands are specified by an operator with a graphical interface [19] and a remote site where the commands are executed [20]. The remote site was developed to be able to

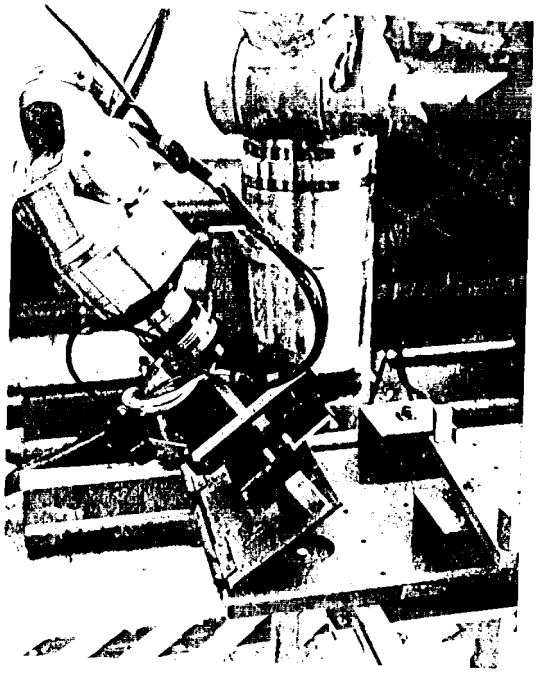

Figure 2: Door opening task

execute multiple concurrent behaviors as described by local site command parameterization, and has been implemented in Ada to be consistent with language constraints for Space Station systems. The system currently uses a seven DOF Robotics Research Corporation K-1207 dextrous manipulator with a six DOF LORD force-torque sensor at the wrist and a servoed gripper. Autonomous commands are generated with the local site system and sent for execution at the remote site. For teleoperation and shared control tasks, the operator uses a six DOF hand controller. The system is implemented in a six CPU 68020/68881 environment and generates joint position commands each 20 ms which are sent to the manufacturer controller which supplies the joint servo control.

6. Results

A door opening task, as shown in Figure 2, is used to demonstrate the use of different control inputs to execute a task. Figures 3 - 8 show experimental results opening the door using shared control, force control, and the spring trajectory. In all three cases the motion frame, where the impedance equation is evaluated, was placed such that its X axis was aligned with the hinge axis. The diagonal M matrix had translational masses of 10 kg and rotational inertias of 2 kg-m^2. The diagonal B matrix had translational gains 350 kg/s and rotational gains of 80 kg-m/s. The diagonal K matrix had translational gains 100 N/m and rotational gains of 10 N/rad.

For shared control door opening, the M and B parameters of the impedance equation were specified and the K matrix was set to zero. The reference force setpoints were all set to zero so that force control would provide compliance to accommodate for inaccuracies in the teleoperated motion of the robot. Tool mode teleoperation of the hand controller was used with the mapping set such that a one DOF rotation of the hand controller wrist was mapped to a rotation about the MOTION frame X axis. The diagonal B_t matrix was set to zero except for the X rotation component so the operator could only specify rotation about the hinge axis. The results of the door opening task using shared control are shown in Figures 3 and 4.

For force control door opening, the M and B parameters of the impedance equation were specified

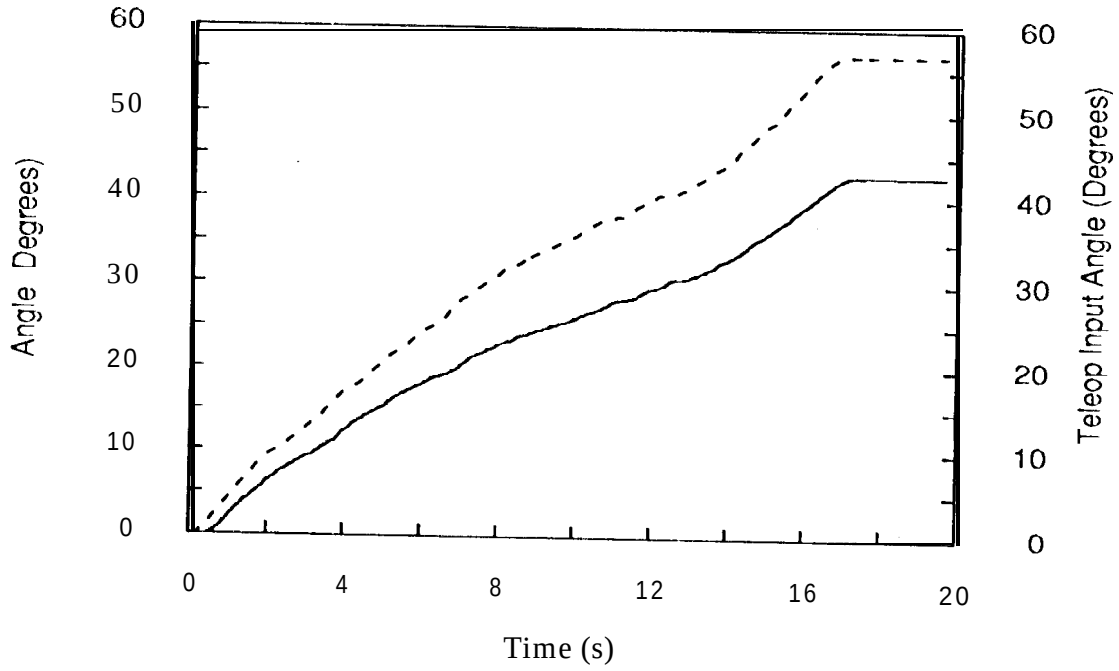

Figure 3: Shared control door opening: solid is door rotation about the MOTION frame X axis (hinge axis); dashed is integrated teleoperation input angle

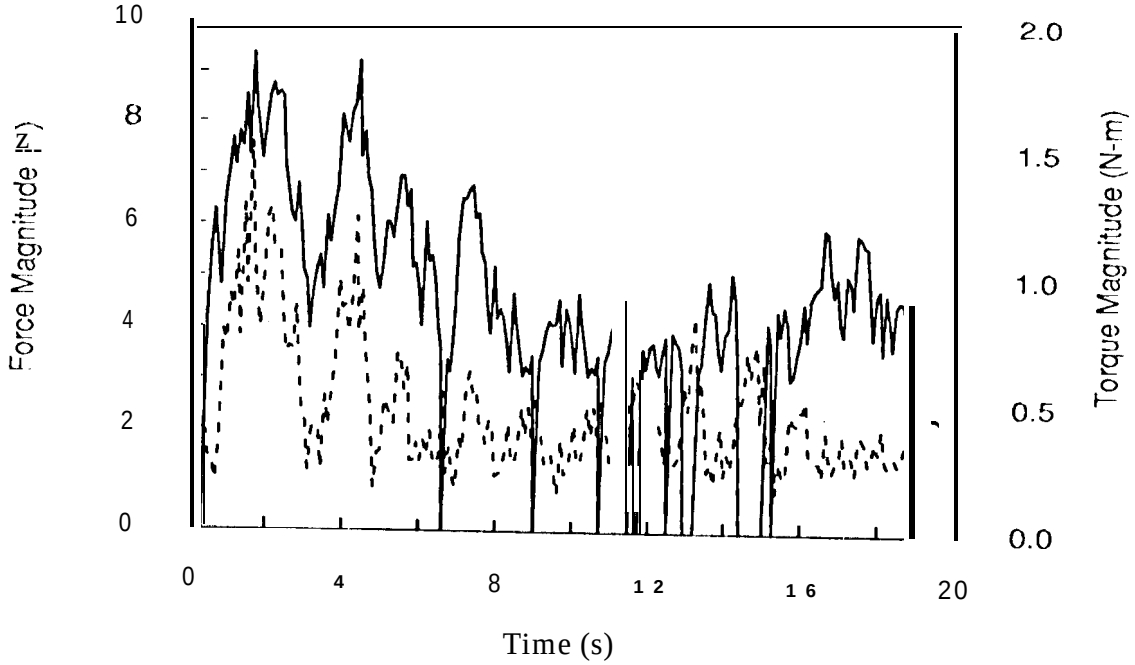

Figure 4: Shared control door opening: solid is force vector magnitude in MOTION frame; dashed is torque vector magnitude in MOTION frame

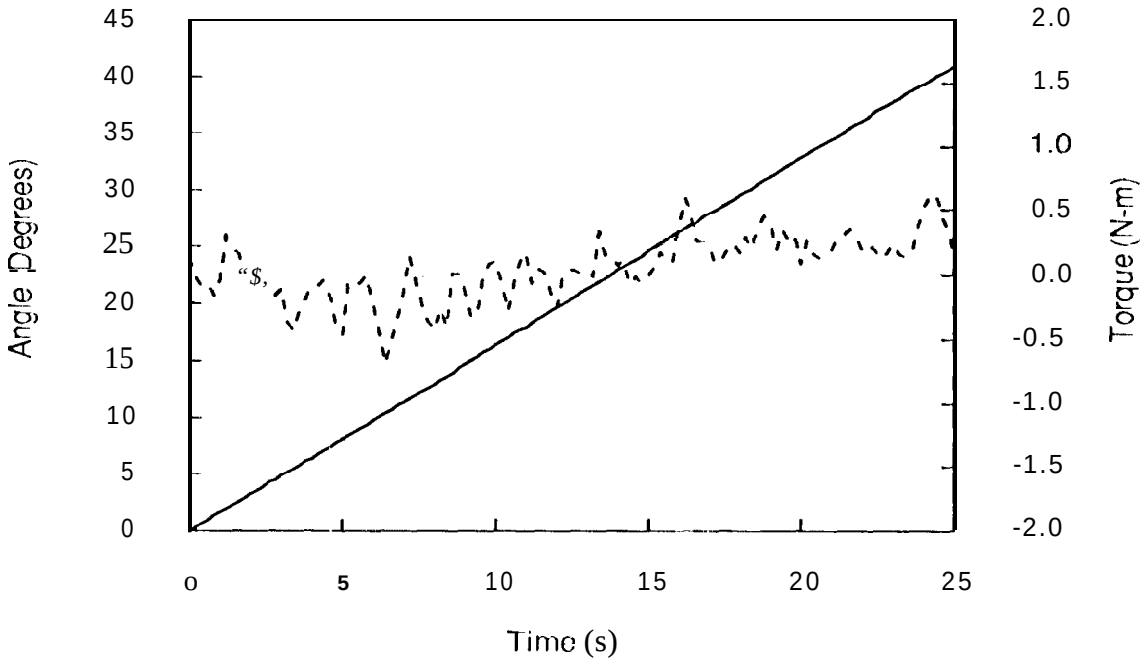

Figure 5: Force control door opening: solid is door rotation about the MOTION frame X axis (hinge axis); dashed is torque about the MOTION frame X axis

and the K matrix was set to zero. The reference force setpoints were all set to zero except for the torque about the X axis which was set to 18 N-m. This torque setpoint caused the door to open. The results of the door opening task using force control are shown in Figures 5 and 6.

For spring trajectory door opening, the M, B, and K parameters of the impedance equation were specified. The reference force setpoints were all set to zero so that force control would provide compliance. The reference position set point X_r was set to the destination position representing a rotation of 35 degrees. The reference velocity and acceleration were set to zero. The results of the door opening task using the spring trajectory are shown in Figures 7 and 8. The sensed force and torque magnitudes could probably have been reduced by setting the K matrix gains to zero except for about the X axis. Inaccuracy in the specified goal position versus the physical position after opening the door could have caused the sensed forces and torques to increase as the door opened.

Conclusions

A control architecture for data driven merging of concurrent control behaviors has been developed and implemented on a redundant manipulator. This data driven approach provides a large suite of available control modes using a large variety of real and virtual sensors within the practical constraint of a single validated software system. The architecture provides a flexible remote site controller that is well suited for supervisory telerobotics applications. The abstraction between task behaviors and manipulator specific mappings allows the approach to be applied to a wide variety of mechanisms with a user defined task parameterization.

Acknowledgements

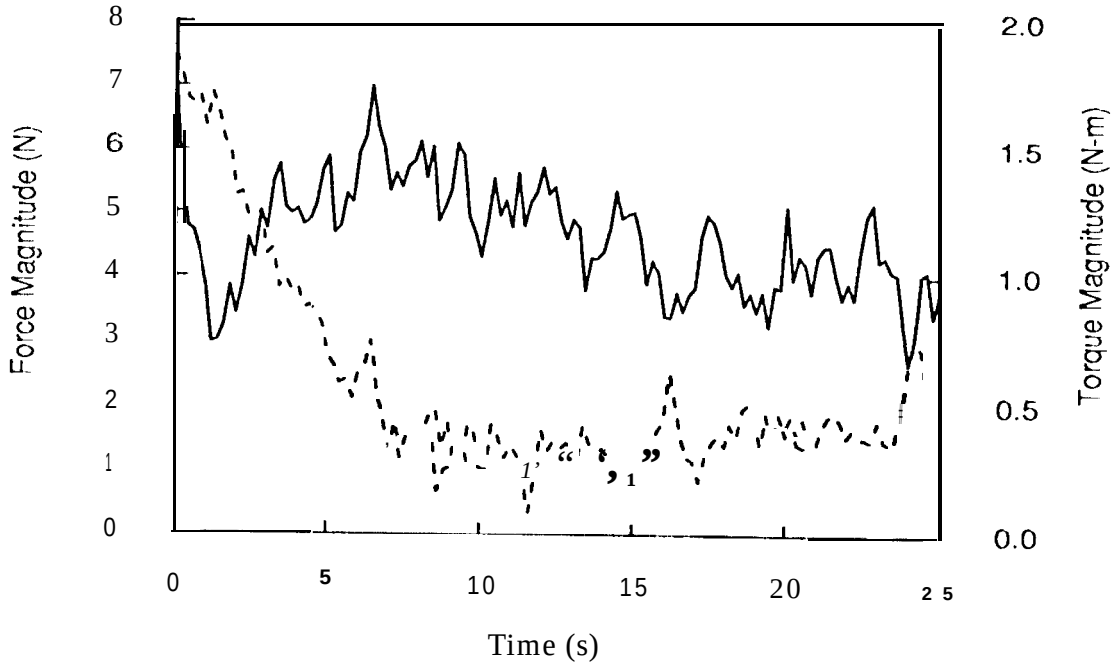

Figure 6: Force control door opening: solid is force vector magnitude in MOTION frame; dashed is torque vector magnitude in MOTION frame

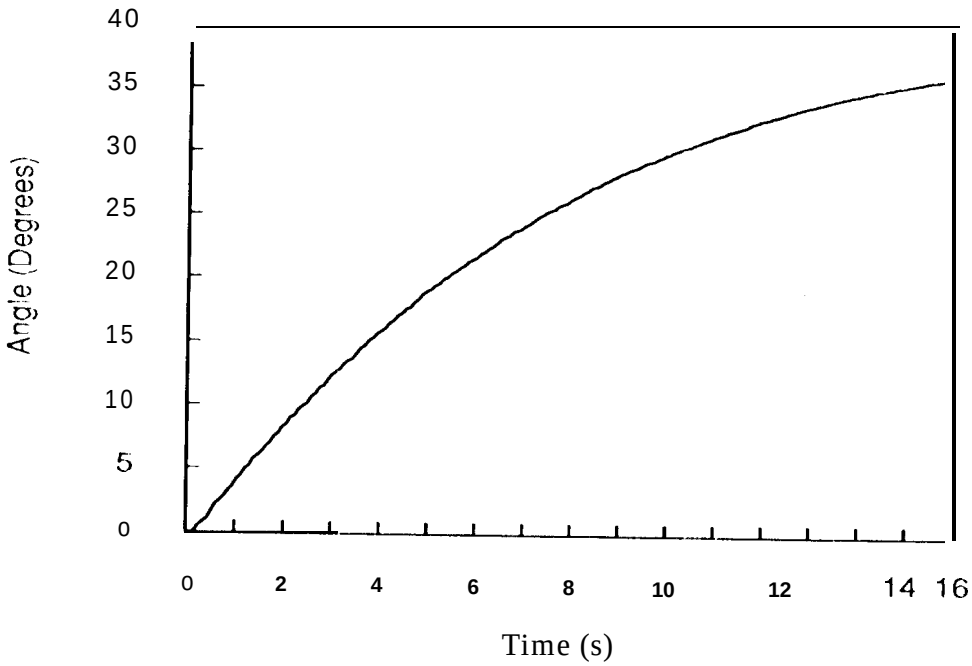

Figure 7: Spring trajectory door opening: door rotation about the MOTION frame X axis (hinge axis)

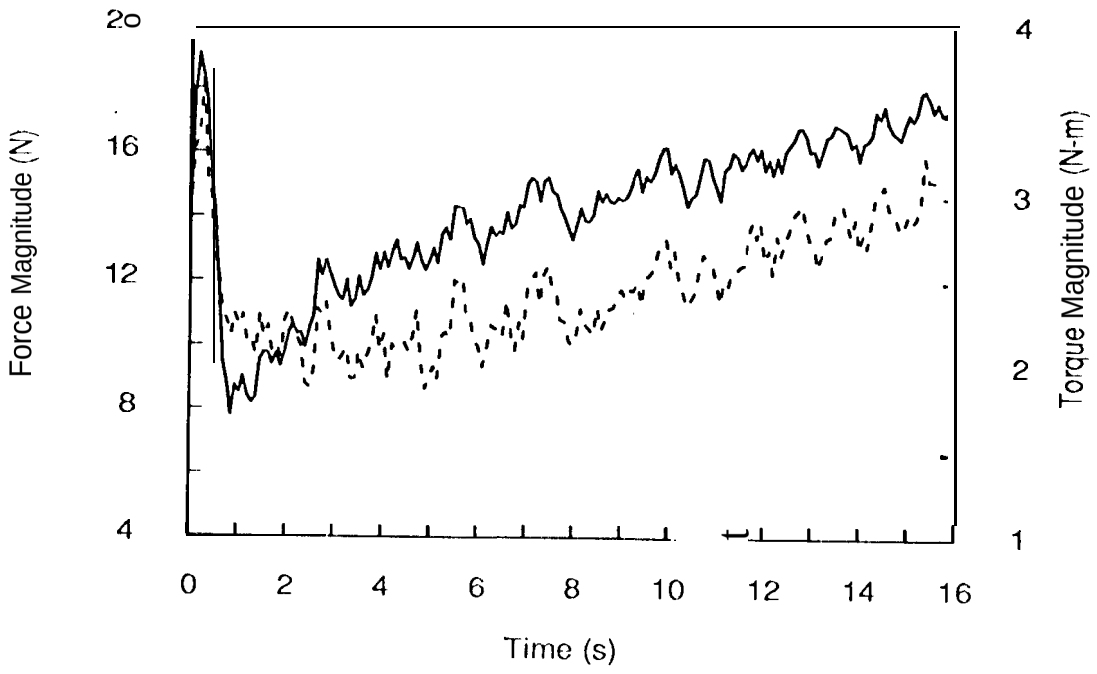

Figure 8: Spring trajectory door opening: solid is force vector magnitude in MOTION frame; dashed is torque vector magnitude in MOTION frame

The research described in this paper was carried out at the Jet Propulsion Laboratory, California nstitute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] H. Seraji, Mark K. Long, and T. Lee. Motion control of 7 dof arms: The configuration control approach. In *IEEE Int. Conf. on Robotics and Automation*, 1991.

[2] Mark K. Long. Task directed inverse kinematics for redundant manipulators. *Journal of Intelligent and Robotic Systems*, November 1992.

[3] Wyatt S. Newman and Mark E. Dohring. Augmented impedance control: An approach to compliant control of kinematically redundant manipulators. In *IEEE Int. Conf. on Robotics and Automation*, 1991.

[4] H. Seraji and R. Colbaugh. Improved configuration control for redundant robots. *Journal of Robotic Systems*, 7(6):897- 928, 1990.

[5] N. Hogan. Impedance control: An approach to manipulation: Part i - theory. *ASME Journal of Dynamic Systems, Measurement, and Control*, 107:1- 7, March 1985.

[6] Dale A. Lawrence and R. Michael Stoughton. Position-based impedance control: Achieving stability in practice. In *Proceedings AIAA Conference on Guidance, Navigation, and Control*, pages 221- 226, Monterey, Ca, August 1987.

[7] Paul G. Backes. Generalized compliant motion task description and execution within a complete telerobotic system. In *Proceedings IEEE International Conference on Systems Engineering*, August 9 11 1990.

[8] Paul G. Backes. Generalized compliant motion with sensor fusion. In *Proceedings 1991 ICAR: Fifth International Conference on Advanced Robotics, Robots in Unstructured Environments*, pages 1281 1286, Pisa, Italy, June 19 22 1991.

[9] R. Colbaugh. A unified approach to control manipulators with redundancy. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 1, pages 1 18, 1990.

[10] H. Seraji, M. Long, and T. Lee. Configuration control of 7 dof arms. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 1195 1200, 1991.

[11] Mark K. Long. Task directed inverse kinematics for redundant manipulators. *Journal of Intelligent and Robotic Systems*, 6:241 261, 1992.

[12] Y. Nakamura and H. Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator cont 1'01. A *SME Journal of Dynamic Systems, Measurement, and Control*, I 08(3): I 63 17 1 , 1986.

[13] C.W. Wampler and L.J. Leifer. Applications of damped least squares methods to 1'(; SOIV('(l rate and resolved acceleration cont 1'01 of manipulat ors. *ASME Journal of Dynamic Systems, Measurement, and Control*, I 1 ()(1):31 38, 1988.

[11] O. Egeland, J.R. Sagli, and I. Spangelo. A damped least-squares solution to redundancy resolut ion. In *Proceedings IEEE International Conference on Robotics and .4 utomation*, Sacramento, ('A. April 1991.

[15] Paul G. Backes. Multi-sensor based impedance control for task execution. In *Proceedings IEEE International Conference on Robotics and Automation*, Nice, France, May 1992.

[16] Richard A. Volpe. Task space velocity blending for real-time trajectory generation. In *Proceedings IEEE International Conference on Robotics and Automation*, Atlanta, Georgia, May 1993.

[17] Mark K. Long. Qualitative singularity analysis of the 7 dof robotics research k-1207. Jet Propulsion Laboratory Engineering Memorandum 317-91-299 (internal document), 1991.

[18] Paul G. Backes. Ground-remote control for space station telerobotics with time delay. In *Proceedings AAS Guidance and Control Conference*, Keystone, CO, February 8-12 1992. AAS paper No. 92-052.

[19] Paul G. Backes, John Beahan, and Bruce Bon. Interactive command building and sequencing for supervised autonomy. In *Proceedings IEEE International Conference on Robotics and Automation*, Atlanta, Georgia, May 1993.

[20] Paul G. Backes, Mark K. Long, and Robot D. Steele. The modular telerobot task execution system for space telerobotics. In *Proceedings IEEE International Conference on Robotics and Automation*, Atlanta, Georgia, May 1993.